"D" Release - Closed Loop Use Case: Simulation of O-RU and O-DU

O1 Simulators for O-DU and O-RU

Scope

The purpose of the O1 Simulators is to provide a NETCONF Server framework exposing the OpenFronth aul November 2020 train models, in the case of O-RU, and the use case specific model for the O-DU. In addition, the Simulators provide the already existing capabilities like NETCONF Call Home (via SSH), sending VES messages (pnfRegistration, fault, file-ready etc.).

Artifacts

Each of the simulators is released as a docker image which can be used independent from any other image.

- O-RU Simulator: nexus3.o-ran-sc.org:10002/o-ran-sc/nts-ng-o-ran-ru-fh:1.3.3
- O-DU Simulator: nexus3.o-ran-sc.org:10002/o-ran-sc/nts-ng-o-ran-du:1.3.3

Usage

The Simulators can be started easily via docker-compose. Example of such a file:

docker-compose.yaml

```
version: '3.8'
services:
  ntsim-ng-o-ru:
    image: "${DOCKER_REPO}nts-ng-o-ran-ru-fh:${NTS_BUILD_VERSION}"
    cap_add:
      - SYS_ADMIN
    stop_grace_period: 5m
    hostname: "O-RAN-O-RU-1"
    ports:
     - "::18300:830"
    environment:
      IPv6_ENABLED: ${IPv6_ENABLED}
      SSH_CONNECTIONS: ${SSH_CONNECTIONS}
      TLS_CONNECTIONS: ${TLS_CONNECTIONS}
      NTS_NF_STANDALONE_START_FEATURES: "datastore-populate netconf-call-
home web-cut-through"
      NTS_NF_MOUNT_POINT_ADDRESSING_METHOD:
${NTS_NF_MOUNT_POINT_ADDRESSING_METHOD}

      NTS_HOST_IP: ${NTS_HOST_IP}
      NTS_HOST_BASE_PORT: ${NTS_HOST_BASE_PORT}
      NTS_HOST_NETCONF_SSH_BASE_PORT: ${NTS_HOST_NETCONF_SSH_BASE_PORT}
      NTS_HOST_NETCONF_TLS_BASE_PORT: ${NTS_HOST_NETCONF_TLS_BASE_PORT}
      NTS_HOST_TRANSFER_FTP_BASE_PORT: ${NTS_HOST_TRANSFER_FTP_BASE_PORT}
      NTS_HOST_TRANSFER_SFTP_BASE_PORT: ${NTS_HOST_TRANSFER_SFTP_BASE_PORT}
      NTS_BUILD_DATE: ${NTS_BUILD_DATE}

      SDN_CONTROLLER_PROTOCOL: ${SDN_CONTROLLER_PROTOCOL}
      SDN_CONTROLLER_IP: ${SDN_CONTROLLER_IP}
      SDN_CONTROLLER_PORT: ${SDN_CONTROLLER_PORT}
      SDN_CONTROLLER_CALLHOME_PORT: ${SDN_CONTROLLER_CALLHOME_PORT}
      SDN_CONTROLLER_USERNAME: ${SDN_CONTROLLER_USERNAME}
      SDN_CONTROLLER_PASSWORD: ${SDN_CONTROLLER_PASSWORD}

      VES_COMMON_HEADER_VERSION: ${VES_COMMON_HEADER_VERSION}
      VES_ENDPOINT_PROTOCOL: ${VES_ENDPOINT_PROTOCOL}
      VES_ENDPOINT_IP: ${VES_ENDPOINT_IP}
      VES_ENDPOINT_PORT: ${VES_ENDPOINT_PORT}
      VES_ENDPOINT_AUTH_METHOD: ${VES_ENDPOINT_AUTH_METHOD}
      VES_ENDPOINT_USERNAME: ${VES_ENDPOINT_USERNAME}
```

```
      VES_ENDPOINT_PASSWORD: ${VES_ENDPOINT_PASSWORD}

  ntsim-ng-o-du:
    image: "${DOCKER_REPO}nts-ng-o-ran-du:${NTS_BUILD_VERSION}"
    cap_add:
      - SYS_ADMIN
    stop_grace_period: 5m
    hostname: "O-RAN-O-DU-1"
    ports:
     - "::18310:830"
    environment:
      IPv6_ENABLED: ${IPv6_ENABLED}
      SSH_CONNECTIONS: ${SSH_CONNECTIONS}
      TLS_CONNECTIONS: ${TLS_CONNECTIONS}
      NTS_NF_STANDALONE_START_FEATURES: "datastore-populate ves-heartbeat
ves-file-ready ves-pnf-registration web-cut-through"
      NTS_NF_MOUNT_POINT_ADDRESSING_METHOD:
${NTS_NF_MOUNT_POINT_ADDRESSING_METHOD}

      NTS_HOST_IP: ${NTS_HOST_IP}
      NTS_HOST_BASE_PORT: ${NTS_HOST_BASE_PORT}
      NTS_HOST_NETCONF_SSH_BASE_PORT: ${NTS_HOST_NETCONF_SSH_BASE_PORT}
      NTS_HOST_NETCONF_TLS_BASE_PORT: ${NTS_HOST_NETCONF_TLS_BASE_PORT}
      NTS_HOST_TRANSFER_FTP_BASE_PORT: ${NTS_HOST_TRANSFER_FTP_BASE_PORT}
      NTS_HOST_TRANSFER_SFTP_BASE_PORT: ${NTS_HOST_TRANSFER_SFTP_BASE_PORT}
      NTS_BUILD_DATE: ${NTS_BUILD_DATE}

      SDN_CONTROLLER_PROTOCOL: ${SDN_CONTROLLER_PROTOCOL}
      SDN_CONTROLLER_IP: ${SDN_CONTROLLER_IP}
      SDN_CONTROLLER_PORT: ${SDN_CONTROLLER_PORT}
      SDN_CONTROLLER_CALLHOME_PORT: ${SDN_CONTROLLER_CALLHOME_PORT}
      SDN_CONTROLLER_USERNAME: ${SDN_CONTROLLER_USERNAME}
      SDN_CONTROLLER_PASSWORD: ${SDN_CONTROLLER_PASSWORD}

      VES_COMMON_HEADER_VERSION: ${VES_COMMON_HEADER_VERSION}
      VES_ENDPOINT_PROTOCOL: ${VES_ENDPOINT_PROTOCOL}
      VES_ENDPOINT_IP: ${VES_ENDPOINT_IP}
      VES_ENDPOINT_PORT: ${VES_ENDPOINT_PORT}
      VES_ENDPOINT_AUTH_METHOD: ${VES_ENDPOINT_AUTH_METHOD}
      VES_ENDPOINT_USERNAME: ${VES_ENDPOINT_USERNAME}
      VES_ENDPOINT_PASSWORD: ${VES_ENDPOINT_PASSWORD}

# We can attach to any other docker network, like in the example
below
# networks:
#   default:
#     external:
#       name: smo_integration
```

The variables which are used in the compose file can be defined separately in a .env file:

.env

```
DOCKER_REPO=nexus3.o-ran-sc.org:10004/o-ran-sc/
NTS_MANAGER_PORT=8300
NTS_BUILD_VERSION=1.3.3

IPv6_ENABLED=true
SSH_CONNECTIONS=1
TLS_CONNECTIONS=0
# NTS_HOST_IP=10.20.11.136
NTS_HOST_IP=2001:db8:1::1
NTS_HOST_BASE_PORT=50000
NTS_HOST_NETCONF_SSH_BASE_PORT=50000
NTS_HOST_NETCONF_TLS_BASE_PORT=52000
NTS_HOST_TRANSFER_FTP_BASE_PORT=54000
NTS_HOST_TRANSFER_SFTP_BASE_PORT=56000
NTS_BUILD_DATE=2021-05-01T08:20:54.9Z

NTS_NF_MOUNT_POINT_ADDRESSING_METHOD=docker-mapping
NTS_NF_STANDALONE_START_FEATURES="datastore-populate ves-heartbeat ves-
file-ready ves-pnf-registration netconf-call-home web-cut-through"

SDN_CONTROLLER_PROTOCOL=http
SDN_CONTROLLER_IP=172.40.0.21
SDN_CONTROLLER_PORT=8181
SDN_CONTROLLER_CALLHOME_PORT=6666
SDN_CONTROLLER_USERNAME=admin
SDN_CONTROLLER_PASSWORD=admin

VES_COMMON_HEADER_VERSION=7.1
VES_ENDPOINT_PROTOCOL=https
VES_ENDPOINT_IP=172.40.0.90
VES_ENDPOINT_PORT=8443
VES_ENDPOINT_AUTH_METHOD=basic-auth
VES_ENDPOINT_USERNAME=sample1
VES_ENDPOINT_PASSWORD=sample1
```

Environment values explanation

Next we will detail the meaning of the variables present in the .env file:

- **DOCKER_REPO**: the repository where the docker images can be found. Currently set to nexus3. If a user builds the images locally, from the source code, this needs to be adjusted accordingly
- **NTS_BUILD_VERSION**: the version to be used
- **IPv6_ENABLED**: controls whether IPv6 is used by the Simulators. If set to true, the Simulator expects the NTS_HOST_IP to be a valid IPv6 address from the host machine
- **SSH_CONNECTIONS**: the number of NETCONF SSH Endpoints to be exposed by the simulated NF. If not doing load and stress test in the SDN Controller, it should be set to 1
- **TLS_CONNECTIONS**: the number of NETCONF TLS Endpoints to be exposed by the simulated NF. Should be 1 if NETCONF over TLS is targeted
- **NTS_HOST_IP**: a valid IPv4 address (if IPv6 flag is disabled) or IPv6 address (if IPv6 flag is enabled) which should be on the host machine, such that the simulated NF can be reached through it. The IP address from the host, which should be used by systems outside the local machine to address the simulators
- **NTS_HOST_BASE_PORT**: the port from where the allocation for the simulated network functions should start, if not specified otherwise separately (see below); any port not defined will automatically be assigned to *BASE_PORT*; **NOTE** that in order for a port to be eligible, it must be greater than or equal to **1000**
    - **NTS_HOST_NETCONF_SSH_BASE_PORT**: base port for NETCONF over SSH
    - **NTS_HOST_NETCONF_TLS_BASE_PORT**: base port for NETCONF over TLS
    - **NTS_HOST_TRANSFER_FTP_BASE_PORT**: base port for FTP (for VES fileReady)
    - **NTS_HOST_TRANSFER_SFTP_BASE_PORT**: base port for SFTP (for VES fileReady)
- **NTS_NF_MOUNT_POINT_ADDRESSING_METHOD**: either "docker-mapping" or "host-mapping"; docker-mapping means that the IP address and port used for addressing the simulated NF will be the docker IP and docker port of the container which is associated with that NF; host-mapping means that the NTS_HOST_IP and NTS_HOST_xxxx_PORT (which are mapped to the host machine) will be used to address that simulated NF (e.g. in pnfRegistration message)

- **NTS_NF_STANDALONE_START_FEATURES**: string with values separated by spaces containing features that should be started at boot time by the simulated NF. Possible values:
    - ***datastore-populate***: the datastore exposed by the simulated NF is pre-populated with read-write and read-only random values, such that the exposed models are not empty from a NETCONF Client perspective
    - ***ves-heartbeat***: the simulated NF is able to send periodically VES heartbeat events, according also to the configuration from this xPath: */nts-network-function:simulation /network-function/ves/heartbeat-period*
    - ***ves-file-ready***: the simulated NF is able to send VES fileReady events, when triggered via this RPC: */nts-network-function:invoke-ves-pm-file-ready*
    - ***ves-pnf-registration***: the simulated NF will send, at boot time, a VES pnfRegistration message
    - ***netconf-call-home***: the simulated NF will try to contact the SDN Controller, via the Call Home port and establish a NETCONF Call Home via SSH connection
    - ***web-cut-through***: the simulated NF will have the GUI Cut-through feature enabled (it means that it exposes a leaf that points to the web craft terminal of the NF - in the case of the simulator, the address of that actually points to the ConfigApp inside SDN-R
    - ***manual-notification-generation***: the simulated NF is able to send NETCONF Notifications, triggered manually by the user using the RPC defined at this xPath: */nts-network-function:invoke-notification* (takes as an input the entire notification as a string, either in XML or JSON format)
- **SDN_CONTROLLER_PROTOCOL**: can be either "http" or "https"
- **SDN_CONTROLLER_IP**: the IP address of the SDN Controller with which the simulated NF will interact (e.g. via NETCONF Call Home)
- **SDN_CONTROLLER_PORT**: the Port of the SDN Controller with which the simulated NF will interact (e.g. for sending REST requests)
- **SDN_CONTROLLER_CALLHOME_PORT**: the Port of the SDN Controller with which the simulated NF will interact, where Call Home is listening for connections (e.g. via NETCONF Call Home)
- **SDN_CONTROLLER_USERNAME**: the username to be used when addressing the SDN Controller
- **SDN_CONTROLLER_PASSWORD**: the password to be used when addressing the SDN Controller
- **VES_COMMON_HEADER_VERSION**: the version to be used in the VES Common Header messages which are being generated
- **VES_ENDPOINT_PROTOCOL**: the protocol to be used when addressing the VES Collector (either "http" or "https")
- **VES_ENDPOINT_IP**: the IP address of the VES Collector
- **VES_ENDPOINT_PORT**: the Port of the VES Collector
- **VES_ENDPOINT_AUTH_METHOD**: the authentication method to be used when trying to reach the VES Collector (supported currently: "no-auth" for no authentication, or "basic-auth" for basic HTTP(S) authentication)
- **VES_ENDPOINT_USERNAME**: the username to be used when authenticating to the VES Collector
- **VES_ENDPOINT_PASSWORD**: the password to be used when authenticating to the VES Collector

NETCONF/YANG interface for controlling the simluation

Each simulated NF exposes a proprietary NETCONF/YANG interface which needs to be used for controlling the simulation at runtime (e.g. sending fault events, enabling VES heartbeat etc.). The YANG tree is shown below:

```
module: nts-network-function
  +--ro info
  |  +--ro build-time?         yang:date-and-time
  |  +--ro version?            string
  |  +--ro started-features?   ntsc:feature-type
  +--rw simulation
     +--rw network-function
     |  +--rw function-type?                   string
     |  +--rw mount-point-addressing-method?   enumeration
     |  +--rw fault-generation!
     |  |  +--rw fault-delay-list* [index]
     |  |  |  +--rw index           uint16
     |  |  |  +--rw delay-period?   uint16
     |  |  +--ro fault-count {faults-status}?
     |  |     +--ro normal?     uint32
     |  |     +--ro warning?    uint32
     |  |     +--ro minor?      uint32
     |  |     +--ro major?      uint32
     |  |     +--ro critical?   uint32
     |  +--rw netconf!
     |  |  +--rw faults-enabled?   boolean
     |  |  +--rw call-home?        boolean
     |  +--rw ves!
     |     +--rw faults-enabled?     boolean
     |     +--rw pnf-registration?   boolean
     |     +--rw heartbeat-period?   uint16
     +--rw sdn-controller
     |  +--rw controller-ip?                       inet:ip-address
     |  +--rw controller-port?                     inet:port-number
     |  +--rw controller-netconf-call-home-port?   inet:port-number
     |  +--rw controller-username?                 string
     |  +--rw controller-password?                 string
     +--rw ves-endpoint
        +--rw ves-endpoint-protocol?      enumeration
        +--rw ves-endpoint-ip?            inet:ip-address
        +--rw ves-endpoint-port?          inet:port-number
        +--rw ves-endpoint-auth-method?   authentication-method-type
        +--rw ves-endpoint-username?      string
        +--rw ves-endpoint-password?      string
        +--rw ves-endpoint-certificate?   string

  rpcs:
    +---x datastore-populate
    |  +--ro output
    |     +--ro status    enumeration
    +---x feature-control
    |  +---w input
    |  |  +---w start-features?   ntsc:feature-type
    |  |  +---w stop-features?    ntsc:feature-type
    |  +--ro output
    |     +--ro status    enumeration
    +---x invoke-notification
    |  +---w input
    |  |  +---w notification-format    enumeration
    |  |  +---w notification-object    string
    |  +--ro output
    |     +--ro status    enumeration
    +---x invoke-ves-pm-file-ready
    |  +---w input
    |  |  +---w file-location    string
    |  +--ro output
    |     +--ro status    enumeration
    +---x clear-fault-counters
       +--ro output
          +--ro status    enumeration
```

Generating VES Faults / NETCONF Alarm Notifications

The simulated NF implements a mechanism for generating predefined alarms - either VES faultNotifications, or NETCONF Notifications, or both.

One can control whether faults are enabled or disabled independently via NETCONF and/or VES, through their respective containers described in the sections above. The YANG **fault-generation** container contains:

- **fault-delay-list** - a list with elements which consists of:
  - *index* (unimportant, but needs to be unique)
  - *delay-period* which represents the number of seconds in between the current fault and the next fault which is being generated

NETCONF notifications are being generated if the following attribute is set to "**true**": */nts-network-function:simulation/network-function/netconf/faults-enabled*

VES faultNotifications are being generated if the following attribute is set to "**true**": */nts-network-function:simulation/network-function/ves/faults-enabled*

Please note that the simulated O-DU does not support NETCONF Notifications, because no notifications are defined in the o-ran-sc-du-hello-world.yang

Example of sending alarms

Example of configuration of a simulated NF which sends both a NETCONF Notification and a VES faultNotification every 60 seconds:

```
<simulation xmlns="urn:o-ran-sc:params:xml:ns:yang:nts:network:function">
        <network-function>
            <mount-point-addressing-method>host-mapping</mount-point-
addressing-method>
        <fault-generation>
                <fault-delay-list>
                    <index>0</index>
                    <delay-period>60</delay-period>
                  </fault-delay-list>
        </fault-generation>
        <netconf>
                  <faults-enabled>true</faults-enabled>
                  <call-home>false</call-home>
        </netconf>
        <ves>
                <faults-enabled>true</faults-enabled>
                  <pnf-registration>false</pnf-registration>
                  <heartbeat-period>0</heartbeat-period>
        </ves>
        <function-type>NTS_FUNCTION_TYPE_O_RAN_O_DU</function-type>
        </network-function>
        <sdn-controller>
                <controller-protocol>http</controller-protocol>
                  <controller-ip>172.40.0.21</controller-ip>
                  <controller-port>8181</controller-port>
                  <controller-netconf-call-home-port>6666</controller-
netconf-call-home-port>
                  <controller-username>admin</controller-username>
                  <controller-password>admin</controller-password>
        </sdn-controller>
        <ves-endpoint>
                  <ves-endpoint-protocol>https</ves-endpoint-protocol>
                  <ves-endpoint-auth-method>no-auth</ves-endpoint-auth-
method>
                  <ves-endpoint-ip>172.40.0.90</ves-endpoint-ip>
                  <ves-endpoint-port>8443</ves-endpoint-port>
                  <ves-endpoint-username>sample1</ves-endpoint-username>
                  <ves-endpoint-password>sample1</ves-endpoint-password>
        </ves-endpoint>
</simulation>
```